# Controlo programado com Arduino

**Prática 5: O Semáforo Básico (Controlo Sequencial Avançado)**

Nas práticas anteriores aprendemos a interagir com botões e a registar estados lógicos. Hoje aplicaremos uma abordagem de engenharia clássica para resolver um sistema sequencial coordenado no tempo: um semáforo de trânsito para veículos. Ligaremos três LEDs (Vermelho, Amarelo e Verde) e programaremos o seu comportamento temporal simulando o ciclo real das ruas da nossa cidade.

## Objetivos de Aprendizagem

Ao finalizar esta sessão, serás capaz de:

- Coordenar múltiplos canais de saída digital de forma síncrona e sequencial.
- Estruturar tempos de atraso diferenciados segundo a importância ou perigo do estado físico controlado.
- Organizar um algoritmo sequencial em bloco fechado evitando estados indefinidos ou cruzamentos perigosos de sinais.
- Praticar a declaração múltipla de variáveis de atribuição de pinos de hardware.

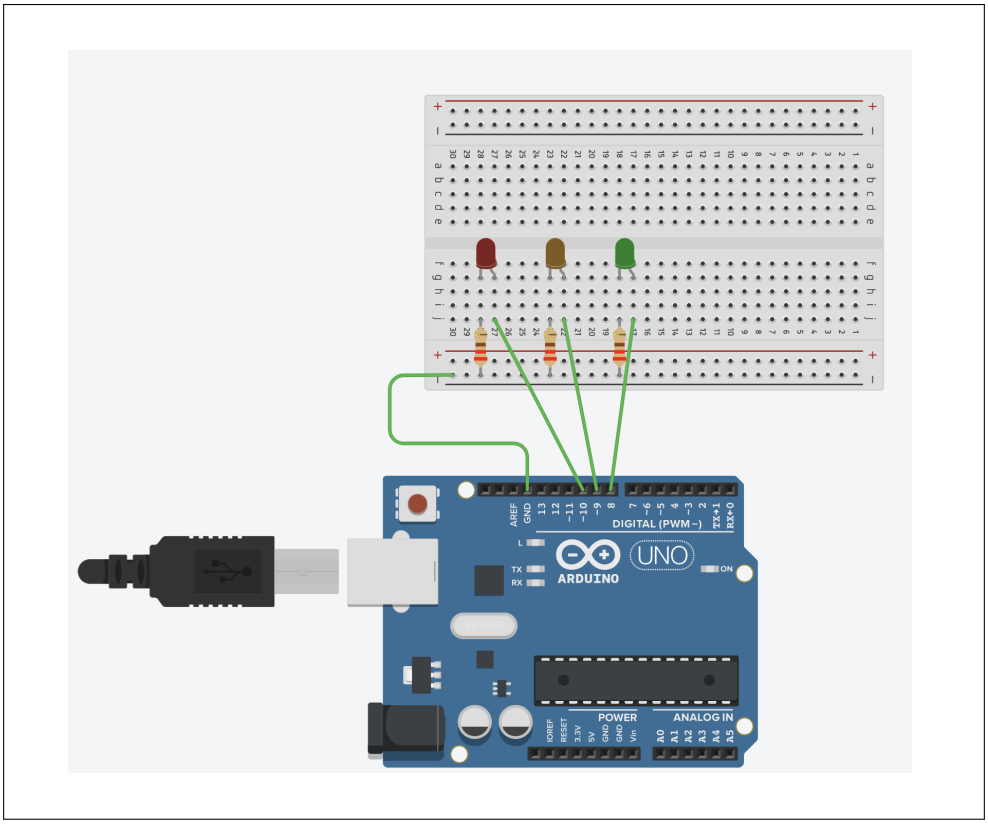
## Componentes Necessários

Procura e coloca os seguintes componentes na tua mesa de trabalho do TinkerCad:

- 1 Placa Arduino Uno.
- 1 Placa de ensaio pequena (Protoboard).
- 3 Díodos LED (cores obrigatórias: 1 Vermelho, 1 Amarelo e 1 Verde).
- 3 Resistências de $220\,\Omega$ (uma para cada LED).
- Cabos de ligação virtuais.

## Esquema de Montagem na Protoboard

Para controlar o semáforo de veículos, associaremos cada cor a um pino digital sequencial do Arduino.



## Código Base: Sequência Temporal Estrita

Escreve o seguinte programa no teu editor de texto do TinkerCad. Observa com atenção como em cada estado do semáforo garantimos o ato de apagar de forma explícita os LEDs que não correspondem para evitar acendimentos simultâneos acidentais.

```
// Declaracao de pinos fisicos para os LEDs do semaforo
int ledVermelho = 10;
int ledAmarelo = 9;
int ledVerde = 8;

void setup() {
  // Configurar os tres pinos como canais de saida
  pinMode(ledVermelho, OUTPUT);
  pinMode(ledAmarelo, OUTPUT);
  pinMode(ledVerde, OUTPUT);
}

void loop() {
  // ESTADO 1: Verde ativo (Passagem livre de veiculos)
  digitalWrite(ledVerde, HIGH);
  digitalWrite(ledAmarelo, LOW);
  digitalWrite(ledVermelho, LOW);
  delay(5000); // O verde dura 5 segundos ativo

  // ESTADO 2: Amarelo ativo (Aviso de paragem iminente)
  digitalWrite(ledVerde, LOW);
  digitalWrite(ledAmarelo, HIGH);
  digitalWrite(ledVermelho, LOW);
  delay(2000); // O amarelo dura apenas 2 segundos

  // ESTADO 3: Vermelho ativo (Veiculos detidos)
  digitalWrite(ledVerde, LOW);
  digitalWrite(ledAmarelo, LOW);
  digitalWrite(ledVermelho, HIGH);
  delay(5000); // O vermelho dura 5 segundos ativo
}
```

Listing 1: Código para a temporização de um semáforo de veículos.

## Como funciona o código?

O fluxo deste programa é puramente sequencial e linear. Ao estar dentro da função `loop`, repete-se infinitamente nesta ordem exata:

1. Acende-se o Verde e apagam-se os restantes $\rightarrow$ Espera de $5$ segundos.
2. Apaga-se o Verde, acende-se o Amarelo e apaga-se o Vermelho $\rightarrow$ Espera de $2$ segundos.
3. Apagam-se os dois anteriores, acende-se o Vermelho $\rightarrow$ Espera de $5$ segundos.
4. O ciclo termina e volta a começar imediatamente no passo 1 (Verde).

**Segurança Elétrica e Lógica**

Em sistemas industriais reais, um erro de software jamais deve permitir que duas luzes contraditórias (como o Verde e o Vermelho) se acendam juntas. Por isso, em cada mudança de fase apagamos explicitamente as saídas inativas através de instruções `digitalWrite(pin, LOW)`.

**O Desafio da Prática 5**

Melhora o comportamento do semáforo aplicando estas variações lógicas sobre o teu código:

1. **Fase de Transição de Segurança**: Em muitos países, antes de passar do semáforo Vermelho para o Verde, acendem-se simultaneamente o Vermelho e o Amarelo durante um segundo para indicar aos condutores que arranquem com os motores. Modifica o teu código para introduzir esta fase intermédia entre o Estado 3 (Vermelho) e o Estado 1 (Verde).

2. **Modo Noturno (Amarelo Intermitente)**: Modifica o código para simular o comportamento de um semáforo à meia-noite: as luzes Verde e Vermelha devem ficar completamente inativas, enquanto a luz Amarela deve acender-se e apagar-se de maneira contínua a cada $500$ ms.

3. *Pergunta para refletir*: Se quisesses que o semáforo fosse interativo e um peão pudesse mudar o ciclo de luzes pressionando um botão, que problemas achas que levantaria o uso da instrução `delay(5000)`? É o Arduino capaz de detetar uma pulsação enquanto está a executar essa pausa temporal?